ETC Installation Procedure

This document describes the ETC installation for CICS-related operating systems, and covers the following major installation tasks:

- Loading ETC from the Installation Tape onto the Disk
- Defining the ETC Environment with the ETCPARM Macro
- Example of the ETCPARM Macro
- Assembling ETCPARM and Linking ETC

For general information on installing ETC, refer to ETC Installion and Operation.

Loading ETC from the Installation Tape onto the Disk

The steps for loading ETC from the installation tape are operating-system-dependent.

- Loading the Installation Tape for OS/390
- Loading the Installation Tape for VSE/ESA

Loading the Installation Tape for OS/390

If you are using System Maintenance Aid (SMA), refer to the SMA documentation (included on the current edition of the Natural documentation CD).

If you are **not** using SMA, follow the instructions below.

This section explains how to:

- Copy data set COPY.JOB from tape to disk.
- Modify this data set to conform with your local naming conventions.

The JCL in this data set is then used to copy all data sets from tape to disk.

If the datasets for more than one product are delivered on the tape, the dataset COPY.JOB contains the JCL to unload the datasets for all delivered products from the tape to your disk.

After that, you will have to perform the individual install procedure for each component.

Step 1 - Copy data set COPY.JOB from tape to disk

The data set COPY.JOB (label 2) contains the JCL to unload all other existing data sets from tape to disk. To unload COPY.JOB, use the following sample JCL:


```
//SAGTAPE JOB SAG,CLASS=1,MSGCLASS=X
//* --------------------------------
//COPY EXEC PGM=IEBGENER
//SYSUT1 DD DSN=COPY.JOB,
// DISP=(OLD,PASS),
// UNIT=(CASS,,DEFER),
// VOL=(,RETAIN,SER=<Tnnnnn>),
// LABEL=(2,SL)
//SYSUT2 DD DSN=<hilev>.COPY.JOB,
// DISP=(NEW,CATLG,DELETE),
// UNIT=3390,VOL=SER=<vvvvvv>,
// SPACE=(TRK,(1,1),RLSE),
// DCB=*.SYSUT1
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//
```

Where:

<*hilev*>	is a valid high level qualifier
<*Tnnnnn*>	is the tape number
<*vvvvvv*>	is the desired volser

Step 2 - Modify COPY.JOB to conform with your local naming conventions

There are three parameters you have to set before you can submit this job:

- Set HILEV to a valid high level qualifier.
- Set LOCATION to a storage location.
- Set EXPDT to a valid expiration date.

Step 3 - Submit COPY.JOB

Submit COPY.JOB to unload all other data sets from the tape to your disk.

Loading the Installation Tape for VSE/ESA:

If you are **not** using Software AG's System Maintenance Aid (SMA), copy the sublibrary containing the object deck, assembler macro and sample parameter module from tape using the following job control. The sample JCS supplied on tape for the installation of ETC assumes one library (SAGLIB):


```
* $$ JOB JNM=RESTORE,CLASS=0
* $$     DISP=D,LDEST=*
* $$ LST CLASS=A,DISP=D
// JOB RESTORE
// ASSGN SYS005,IGN
// ASSGN SYS006,CUU,VOL=volser
// MTC REW,SYS006
// MTC FSF,SYS006,nn
* *** Now process ETCnnn.LIBR - JOBS ***
// EXEC LIBR,PARM='MSHP'
 RESTORE SUB=SAGLIB.ETCnnn:SAGLIB.ETCnnn -
         TAPE=SYS006 -
         LIST=YES -
         REPLACE=NO
/*
/&
* $$ EOJ
```

With the VOL parameter, replace *volser* with the volume serial number of the tape. For the value of *nn*, see the **Report of Tape Creation**.

Defining the ETC Environment with the ETCPARM Macro

Create the ETC parameter module ETCPARM by coding the ETCPARM macro with appropriate parameters in an assembly file. The member SAMPLE contains an example which you should change, according to your installation requirements. The ETCPARM macro call begins with the specification of one or more database IDs as positional parameters, followed by optional keyword parameters.

The following ETCPARM parameters are available:

dbid | ADANAME | ADMIN | AMODE31 | ASYNC | FBOPT | PUSERS | PUSERTO | SAP | STCK | TIMEOUT | TRNAME

dbid - Define Database as ETP Database

```
ETCPARM dbid
```

Specify one or more database IDs that are to be defined as ETP databases.

```
ETCPARM dbid,dbid,...
```

Multiple database IDs must be separated by commas.

```
ETCPARM *
```

To define all databases as ETP databases, specify "*" as the only positional parameter. This parameter must be specified. Valid database IDs are 1 - 65535, except 255.

```
ETCPARM (fromdbid,todbid)
```

When the format *(from dbid, to dbid)* is specified, the first database ID *from dbid* is translated into the second *to dbid* if the *from dbid* is encountered in an Adabas control block. The first database ID *from dbid* in this configuration can be zero (0).

Multiple pairs of the format *(from dbid, to dbid)* must be separated by commas. This format can be mixed with the format that specifies *dbid* only. Up to 512 of these pairs and/or *dbids* may be specified.

To limit overhead, Software AG strongly recommends specifying only those databases containing a master file defined in the administration file (see ADMIN parameter, below). This enables a function similar to that provided by the Natural NTDB macro.

ADANAME - Specify Routine for Handling Adabas Calls

Syntax:

```
ADANAME=routinename
```

- where *routinename* is the name of the routine (default is ADALNC) to which control is passed for handling Adabas calls.

ADMIN - Specify Database ID and File Number of Administration File

Syntax:

```
ADMIN=(dbid, file [, <password [, ciphcode ] ])
```

Specify the database ID and file number of the administration file.

If required, the administration file's password and/or cipher code must also be specified.

AMODE31 - Enable/Disable 31-Bit Addressing Mode

Syntax:

```
AMODE31=value
```

Possible Values:

AMODE31=NO	Setting AMODE31=NO prevents ETC from switching to 31-bit addressing mode, even if such addressing is possible.
AMODE31=YES	This setting (default) allows the switch to 31-bit mode, if possible, and permits acquiring storage above the 16-MB storage line.

Specify NO only if any of the following are true:

- An application passes 24-bit addresses with the high-order (leftmost) byte is neither X'00' nor X'80'.
- An ADALNC module is being used which has been assembled and linked in 24-bit mode. Note, however, that reassembling and relinking in 31-bit mode is recommended.

For operating-system-specific information, see the *Adabas Installation Manual*.

Generally, COBOL compilers generate correct 24- and 31-bit addresses.

ASYNC - Specify Routine Starting Async Task

Syntax:

```
ASYNCH=asynchname
```

- where *asynchname* is the name of a routine that starts an asynchronous task. The routine is addressed by a V-type address constant generated in the macro expansion, and must follow standard linking conventions.

If ASYNCH is not specified and a master file definition requires the starting of an asynchronous task following and end-of-transaction (ET), an error occurs.

FBOPT - Provide Format Buffer Optimization

Syntax:

```
FBOPT={(ALL,minfbl,num_entry[,timeout])}
FBOPT={(SAP,minfbl,num_entry[,timeout])}
FBOPT={(NO,minfbl,num_entry[,timeout])}
```

Provide Format Buffer optimization for large buffers if GFIDs are used extensively and Adabas has been tuned so that almost no overwrites occur. Optimization is performed only if the database calls are routed through ETP - that is, the database is defined as an ETP database in the ETCPARM macro invocation (see above).

Possible Values:

Value	Explanation
SAP	If the SAP R/2 application system is installed and optimization is wanted for SAP call only, specify SAP as the first subparameter.
ALL	For optimization of calls, specify ALL.
NO	If you do not use the SAP R/2 application system, specify FBOPT=NO (the default value).
minfbl	If either SAP or ALL are specified, the *minfbl* value specifies the minimum Format Buffer length and can be any value ranging 16-1024.
num entry	The *num entry* value specifies the number of 16-byte entries in the table holding global format IDs, and can range 64-32767. If this value is specified too low, optimization becomes ineffective.
timeout	The *timeout* value specifies the time period after which the table entries defined by the *minfbl* and *num entry* that have not been accessed, are deleted. The value is measured in minutes, and defaults to the value of the TIMEOUT parameter specified below.

PUSERS - Specify Number of Users for Parallel Adabas Call Execution

Syntax:

```
PUSERS=value
```

PUSERS specifies the number of users that can execute Adabas calls in parallel. For each user, a slot of 256 bytes of storage is allocated.

value - Valid values are 100 - 99999. The default is 100.

A slot is only used for the time required to process an Adabas call (and all calls that may possibly be issued by ETP). The slot is then marked as being free to be reused by another user. Therefore, the default value of 100 should be sufficient for most installations. If no free slot is available, the task is abended with the ABEND code ETCB. In such a case, the value of this parameter should be increased in steps of 100. PUSERS limits the number of taks executing in ETCNUC in parallel; it does not limit the number of Adabas or CICS tasks that can be handled by ETCNUC.

PUSERTO - Specify Timeout Before Releasing PUSER Slot

Syntax:

```
PUSERTO=value
```

PUSERTO specifies the time, in minutes, after which a PUSERS slot is marked as being free if it has not already been released (e.g. as a result of a user ABEND).

value - The time specified should be greater than the time required to process an Adabas call. Valid values are 1 - 60. The default is 3.

SAP - Specify ADABAS Support for SAP R/2 Application

Syntax:

```
SAP=value
```

Possible Values:

SAP=YES	Specify YES if Adabas support is required for the SAP R/2 application system.
SAP=NO	If you do not use the SAP R/2 application system, enter SAP=NO (the default).

Note: SAP R/2 versions that use the direct call interface without using the CICS TWA are not supported. In addition, SAP R/2 users must request a zap from SAP support that disables SAP R/2 Format Buffer optimization.

STCK - Optional Routine for Substituting Direct STCK Instruction

Syntax:

```
STCK=routinename
```

routinename - Specify the name of the optional routine that substitutes direct Store Clock (STCK) instructions. The routine is addressed by a V-type address constant generated in the macro expansion, and must follow standard linking conventions.

TIMEOUT - Set Time Before Releasing User Work Storage

Syntax:

```
TIMEOUT=value
```

If a user does not perform any ETP actions during the number of minutes specified by TIMEOUT, ETP releases the user's work storage.

value - The TIMEOUT value must be equal to or greater than the total time specified by the Adabas transaction/non-activity (TT, TNAA, TNAE and TNAX) time limits specified by ADARUN. After one of those limits has expired, Adabas issues an implicit Back out Transaction (BT) for all open transactions (TT limit) and deletes the related User Queue elements.

TRNAME - Specify Names of CICS Transactions for Further Processing

Syntax:

```
TRNAME=transname
TRNAME=(transname,...)
```

transname - The name(s) of one or more CICS transactions for which further ETP actions should be performed. To reduce overhead, transactions that are not specified by TRNAME are not processed further. By default, all transactions are enabled for ETP processing.

Example of the ETCPARM Macro

The following is an example of the ETCPARM macro:

```
ETCPARM  2,(0,30),1470,ADMIN=(4,397),TIMEOUT=60,TRNAME=MYTRANS
```

- which would set the following definitions:

- Databases 2 and 1470 are defined as ETP databases. If the DBID in an Adabas control block is zero, it is translated to 30 (the actual database 30 is not defined as an ETP database, but the master files defined in the administration file must be defined with a DBID of 30, not 0);
- File 397 on database 4 is defined as the administration file;
- User storage for a user that remains inactive for 60 minutes is released;
- The routine for handling Adabas calls is named ADALNC (the default);
- The only transaction allowed to process further is MYTRANS.

Assembling ETCPARM and Linking ETC

This section describes how to perform the operating-system-dependent assembly and linkage to integrate ETC into the run-time library.

- Assembling ETCPARM and Linking ETC for OS/390
- Assembling ETCPARM and Linking ETC for VSE/ESA

Assembling ETCPARM and Linking ETC for OS/390

- Using SMA
- Without Using SMA

Using SMA

When using SMA to assemble ETC (SMA job I070, step 5311), the source library for the generated members must be:

SAGLIB.ETC*nnn*.SRCE(SAMPLE)

The output library must be:

SAGLIB.SMALOAD (the name of the resulting module must be ETCPARM)

When using SMA to link ETC (SMA job I080, step 5311), the required input and output library and module definitions are:

```
ETPLIB   DD  DSN=SAGLIB.ETPnnn.LOAD,DISP=SHR
ETCLIB   DD  DSN=SAGLIB.ETCnnn.LOAD,DISP=SHR
SMALIB   DD  DSN=SAGLIB.SMALOAD,DISP=SHR
TPSLIB   DD  DSN=CICS.LOADLIB,DISP=SHR
SYSLMOD  DD  DSN=cics.loadlib,DISP=SHR
     .           .
     .           .
INCLUDE  TPSLIB(DFHEAI)
CHANGE   ETCNUC(adaname)
INCLUDE  ETCLIB(ETCNUC)
INCLUDE  SMALIB(ETCPARM)  (parameter module)
INCLUDE  ETPLIB(ETPNUC)
INCLUDE  TPSLIB(DFHEAI0)
ENTRY    adaname
NAME     adaname(R)
```

- where:

cics.loadlib	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
adaname	Is the name of the final load module. 3GL programs and Natural use this name to perform Adabas calls for Natural. Specify this name in the ADANAME parameter in the Natural parameter module or dynamically at Natural startup.

Without Using SMA

When not using SMA, use the following JCL to generate the ETCPARM module and link ETCNUC, ETCPARM and the module ETPNUC into a single module:

```
//  jobcard
//ALLOC  EXEC PGM=IEFBR14
//SYSLIN   DD DSN=&&OBJ,SPACE=(TRK,(10,10,2),RLSE),UNIT=SYSDA,
//            DISP=(NEW,PASS),
//            DCB=(BLKSIZE=3040,LRECL=80,RECFM=FB,BUFNO=1)
//*
//HLASM    EXEC PGM=ASMA90,
//   PARM='TERM,OBJ,USING(NOLIMIT,MAP,WARN(15))'
//SYSLIB   DD DISP=SHR,DSN=SAGLIB.ETCnnn.SRCE
//SYSUT1   DD DSN=&SYSUT1,SPACE=(1024,(120,120),,,ROUND),UNIT=VIO,
//            DCB=BUFNO=1
//SYSPUNCH DD DUMMY
//SYSTERM  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=&&OBJ(ETCPARM),DISP=(MOD,PASS)
//SYSIN    DD DSN=SAGLIB.ETCnnn.SRCE(SAMPLE)
//*
//LKED   EXEC PGM=HEWL,
// PARM='XREF,LET,LIST,REUS,NORENT,SIZE=(768K,128K),NCAL',
//            REGION=512K,COND=(0,LT)
//SYSLMOD   DD DISP=SHR,DSN=cics.loadlib
//SYSLIN    DD DDNAME=SYSIN
//SYSUT1    DD DSN&&AUTO,UNIT=3380,SPACE=(TRK,(100,10)),
//             DISP=(NEW,DELETE,DELETE)
//SYSPRINT  DD SYSOUT=*
//SYSLIB    DD DSN=&&OBJ,DISP=(OLD,PASS)
//CICSLIB   DD DSN=cics.syslib,DISP=SHR
//ETPLIB    DD DSN=SAGLIB.ETPnnn.LOAD,DISP=SHR
//ETCLIB    DD DSN=SAGLIB.ETCnnn.LOAD,DISP=SHR
//SYSIN     DD *
 INCLUDE CICSLIB(DFHEAI)
 CHANGE  ETCNUC(adaname)
 INCLUDE ETCLIB(ETCNUC)
 INCLUDE SYSLIB(ETCPARM)
 INCLUDE ETPLIB(ETPNUC)
 INCLUDE CICSLIB(DFHEAI0)
 ENTRY adaname
 MODE RMODE(ANY)
 MODE AMODE(31)
 NAME adaname(R)
//
```

- where:

cics.syslib	Contains the modules DFHEAI and DFHEAIO. These modules are required for command-level linkage only.
cics.loadlib	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
adaname	The name of the final load module. 3GL programs and Natural use this name to perform Adabas calls for Natural. Specify this name for the ADANAME parameter in the Natural parameter module or dynamically at Natural startup.

Assembling ETCPARM and Linking ETC for VSE/ESA

Using the following job control, generate the ETCPARM module and link ETCNUC, ETCPARM and the ETP module ETPNUC into a single module:

```
// JOB ASSEMBLE
// OPTION CATAL
// LIBDEF *,SEARCH=searchlibs,CATALOG=cics.loadlib
 PHASE adaname,*
 MODE RMODE(ANY),AMODE(31)
 INCLUDE DFHEAI
// EXEC PGM=ASSEMBLY
        COPY ETCPARM
        COPY SAMPLE
/*
 INCLUDE ETCNUC
 INCLUDE ETPNUC
 INCLUDE DFHEAI0
 INCLUDE
 ENTRY ETCNUC
/*
// EXEC LNKEDT
/&
```

- where:

searchlibs	A list of libraries containing the ETP load module and, if command-level linkage is desired, modules DFHEAI and DFHEAI0.
cics.loadlib	Contains the final load module. This library must be concatenated to DD name DFHRPL to permit CICS to load the module and to find it before any other module with the same name.
adaname	Is the name of the final load module. 3GL programs and Natural use this name to perform Adabas calls. Specify this name for the Natural profile parameter ADANAME in the Natural parameter module or dynamically at Natural startup.

Identifying ETC to CICS

To identify ETC to CICS, specify the following entry in the CICS PPT for the load module that results from the linkage step:

```
DFHPPT  TYPE=ENTRY,PROGRAM=adaname,RES=YES,PGMLANG=ASSEMBLER
```

- where *adaname* is the name of the load module after completing the linkage step (see the operating-system-specific installation information, above). It is not necessary to link the resulting module *adaname* to either your 3GL applications, to Natural or to the ADALNC module.

The parameter value

```
EXECKEY(USER)
```

must be set in the CICS program definition.

